

A Nonlinear Krylov Accelerator for the Boltzmann k-Eigenvalue Problem

Matthew T. Calef*, Erin D. Fichtl, James S. Warsa, Markus Berndt, Neil N. Carlson

Computational Physics and Methods, Los Alamos National Laboratory, Los Alamos, NM 87545-0001

Abstract

We compare variants of Anderson Mixing with the Jacobian-Free Newton-Krylov and Broyden methods applied to the k-eigenvalue formulation of the linear Boltzmann transport equation. We present evidence that one variant of Anderson Mixing finds solutions in the fewest number of iterations. We examine and strengthen theoretical results of Anderson Mixing applied to linear problems.

1. Introduction

The k-eigenvalue formulation of the linear Boltzmann transport equation is widely used to characterize the criticality of fissioning systems [1, 2]. Physically, the largest eigenvalue, generally denoted by k , is the effective neutron multiplication factor that, in the equation, scales the fission production term to achieve a steady-state solution. The corresponding eigenmode describes the neutron flux profile for that steady-state (i.e., critical) system and, when the system is close to criticality, provides useful information about the distribution of the neutron population in space and velocity [1]. Mathematically, the equation is a standard eigenproblem for which power iteration is well-suited because the eigenmode of interest is most commonly that with the largest magnitude [3]. For the deterministic k-eigenvalue problem each step of a true power iteration incurs a heavy computational cost due to the expense of fully inverting the transport operator, therefore a nonlinear fixed point iteration is generally employed in which an *approximate* inversion of this operator is performed at each iteration. In addition to power iteration, the Implicitly Restarted Arnoldi Method has been applied to this problem and has the advantage of being able to compute additional eigenmodes [4]. However, the transport matrix *must* be fully inverted at each iteration, diminishing its computational efficiency and attractiveness when only the dominant eigenmode is desired.

Recently, more sophisticated nonlinear iteration methods employing approximate inversion, predominantly Jacobian-Free Newton-Krylov (JFNK), have been applied with great success [5, 6, 7]. However, there has not yet been a comprehensive comparison of nonlinear solvers. This paper presents such a comparison, examining the performance of three nonlinear solvers—JFNK, Broyden’s Method and Anderson Mixing—applied to the k-eigenvalue problem. A variant of Anderson Mixing [8], first described in [9], is of particular interest because, in the experience of the authors, it is frequently computationally more efficient than JFNK and Broyden’s method.

JFNK is an inexact Newton’s method in which the inversion of the Jacobian is performed to arbitrary precision using a Krylov method (most commonly GMRES) and the Jacobian itself is never formed, but rather its action is approximated using finite differences of arbitrarily close state data. JFNK can be expected to converge quadratically in a neighborhood containing the solution (cf. [10] and the references therein). Each iteration of JFNK requires a nested ‘inner’ iteration and the bulk of the computational effort is expended in this ‘inner’ Krylov inversion of the Jacobian at each ‘outer’ Newton step. At the end of each inversion, the accumulated Krylov space is discarded even though the Jacobian is expected to change minimally during the final Newton steps when a similar space will be rebuilt in the next Newton iteration. In effect, at the end of each iteration, JFNK discards information that may be of use in successive iterations.

*Corresponding author

Email addresses: mcalef@lanl.gov (Matthew T. Calef), efichtl@unm.edu (Erin D. Fichtl), warsa@lanl.gov (James S. Warsa), berndt@lanl.gov (Markus Berndt), nnc@lanl.gov (Neil N. Carlson)

In its standard formulation Broyden's method (cf. [11]), like low memory BFGS (cf. [12]), uses differences in state from successive iterations to make low rank updates to the Jacobian. The Sherman-Morrison-Woodbury update rule is then used to compute the action of the inverse of the Jacobian after such an update. While Broyden's method is restricted to low-rank updates, it provides an explicit representation of the Jacobian allowing one to employ the Dennis-Moré condition [13] to show that it converges super-linearly in a neighborhood containing the solution. Further, it has been shown to solve linear problems of size N in at most $2N$ iterations (cf. [11] and the references therein.)

Anderson Mixing [8] uses differences in state from successive iterations to infer information about the inverse of the Jacobian, which is assumed to be roughly constant in a neighborhood containing all the iterates. Unlike Broyden's method, the updates can be of arbitrary rank. Recent results by Walker and Ni [14] show that, with mild assumptions, Anderson Mixing applied to a linear problem performs as well as the generalized minimum residual method (GMRES) [15]. In this regard, Anderson Mixing may be thought of as a nonlinear version of GMRES. In independent work, Carlson and Miller formulated a so-called nonlinear Krylov acceleration [9] method which we show to be a variant of Anderson Mixing. Further, we examine the hypotheses of a central theorem presented by Walker and Ni and argue that they will, with high probability, be met and that they can be omitted entirely if one is willing to accept a small performance penalty. While Anderson Mixing performs best for our numerical experiments, there is no theory that the authors of this paper know of that can characterize its performance for nonlinear problems.

The rest of this paper is organized as follows: In Section 2 we discuss some of the analysis of Anderson Mixing, motivate an implementation choice and elaborate on the results of Walker and Ni in [14]. In Section 3 we describe the k -eigenvalue problem. Our numerical results can be found in Section 4 and we conclude with an overview.

2. Background of Anderson Mixing and Nonlinear Krylov Acceleration

2.1. Nonlinear Krylov Acceleration

In [9, 16] Carlson and Miller outlined an iterative method, dubbed nonlinear Krylov acceleration or NKA, for accelerating convergence of fixed-point iteration by using information gained over successive iterations. The problem they consider is to find a root of the function $\mathbb{R}^N \ni \mathbf{x} \rightarrow f(\mathbf{x}) \in \mathbb{R}^N$. One approach is to apply a fixed-point iteration of the form

$$\mathbf{x}_{n+1} = \mathbf{x}_n - f(\mathbf{x}_n). \quad (1)$$

Fixed-point iterations can be viewed in the context of an approximate Newton iteration where the derivative of f , denoted Df , is replaced by the identity I . The motivation behind NKA is instead to approximate Df using information from previous iterates, improving that approximation over successive iterations, and in cases where no applicable approximation is available, revert to a fixed-point iteration where Df is replaced with I .

NKA requires an initial guess \mathbf{x}_0 and at the n^{th} invocation provides an update \mathbf{v}_{n+1} that is used to derive the $n + 1^{\text{st}}$ iterate from the n^{th} . This method may be written as

$$\begin{aligned} \mathbf{v}_{n+1} &= NKA[f(\mathbf{x}_n), \dots] \\ \mathbf{x}_{n+1} &= \mathbf{x}_n - \mathbf{v}_{n+1}, \end{aligned}$$

where $NKA[f(\mathbf{x}_n), \dots]$ is the update computed by the nonlinear Krylov accelerator. We use the brackets and ellipsis to indicate that NKA is stateful and draws on previous information when computing an update.

On its first invocation ($n = 0$) NKA has no information and simply returns $f(\mathbf{x}_0)$. At iteration $n > 0$ it has access to the M vectors of differences for some natural number $M \in (0, n)$,

$$\mathbf{v}_i = \mathbf{x}_{i-1} - \mathbf{x}_i \quad \text{and} \quad \mathbf{w}_i = f(\mathbf{x}_{i-1}) - f(\mathbf{x}_i) \quad \text{for } i = n - M + 1, \dots, n,$$

and where, for convenience, we shall define \mathcal{W}_n to be the span of the \mathbf{w}_i vectors available at iteration n . If f has a constant and invertible derivative, Df , then we would have

$$Df\mathbf{v}_i = \mathbf{w}_i \quad \text{and} \quad Df^{-1}\mathbf{w}_i = \mathbf{v}_i. \quad (2)$$

We denote by $\mathcal{P}_{\mathcal{W}_n}$ the operator that projects orthogonally onto the subspace \mathcal{W}_n and write the identity

$$f(\mathbf{x}_n) = \mathcal{P}_{\mathcal{W}_n} f(\mathbf{x}_n) + (f(\mathbf{x}_n) - \mathcal{P}_{\mathcal{W}_n} f(\mathbf{x}_n)).$$

Note that $f(\mathbf{x}_n) - \mathcal{P}_{\mathcal{W}_n} f(\mathbf{x}_n)$ is orthogonal to \mathcal{W}_n . If the \mathbf{w}_i vectors are linearly independent, then there is a unique set of coefficients $\mathbf{z}^{(n)} := (z_1^{(n)}, z_2^{(n)}, \dots, z_n^{(n)}) \in \mathbb{R}^n$ so that

$$\mathcal{P}_{\mathcal{W}_n} f(\mathbf{x}_n) = \sum_{i=n-M+1}^n z_i^{(n)} \mathbf{w}_i,$$

and hence by Eq. (2)

$$Df^{-1} \mathcal{P}_{\mathcal{W}_n} f(\mathbf{x}_n) = Df^{-1} \sum_{i=n-M+1}^n z_i^{(n)} \mathbf{w}_i = \sum_{i=n-M+1}^n z_i^{(n)} \mathbf{v}_i.$$

The idea motivating Carlson and Miller is to project $f(\mathbf{x}_n)$ onto \mathcal{W}_n (the space where the action of Df^{-1} is known), compute that action on the projection and, for lack of information, apply a fixed-point update given in Eq. (1) on the portion of $f(\mathbf{x}_n)$ that is orthogonal to \mathcal{W}_n . The resulting formula for \mathbf{x}_{n+1} is

$$\mathbf{v}_{n+1} = \left[\sum_{i=n-M+1}^n z_i^{(n)} \mathbf{v}_i + \left(f(\mathbf{x}_n) - \sum_{i=n-M+1}^n z_i^{(n)} \mathbf{w}_i \right) \right],$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \mathbf{v}_{n+1},$$

where the vector of coefficients in the orthogonal projection, $\mathbf{z}^{(n)}$, is the solution to the projection, alternatively minimization, problem

$$\mathbf{z}^{(n)} = \arg \min_{\mathbf{y} \in \mathbb{R}^M} \left\| f(\mathbf{x}_n) - \sum_{i=n-M+1}^n y_i \mathbf{w}_i \right\|_2.$$

2.2. NKA as Anderson Mixing

Carlson and Miller had essentially rediscovered, albeit in a slightly different form, the iterative method presented much earlier by Anderson in [8], which is now commonly referred to as Anderson Mixing or Anderson Acceleration. Anderson was studying the problem of finding a fixed point of some function $\mathbb{R}^N \ni \mathbf{x} \rightarrow G(\mathbf{x}) \in \mathbb{R}^N$. In Section 4 of [8] he defines a fixed point residual for iteration i as

$$\mathbf{r}_i = G(\mathbf{x}_i) - \mathbf{x}_i,$$

which can be used to measure how \mathbf{x}_i fails to be a fixed point. With this Anderson proposed an updating scheme of the form¹

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left[\sum_{i=1}^M \tilde{z}_i^{(n)} (\mathbf{x}_n - \mathbf{x}_{n-i}) - \beta_n \left(\mathbf{r}_n - \sum_{i=1}^M \tilde{z}_i^{(n)} [\mathbf{r}_n - \mathbf{r}_{n-i}] \right) \right]$$

where the vector of coefficients, $\tilde{\mathbf{z}}^{(n)} \in \mathbb{R}^M$, is chosen to minimize the quantity

$$\left\| \mathbf{r}_n - \sum_{i=1}^M \tilde{z}_i^{(n)} [\mathbf{r}_n - \mathbf{r}_{n-i}] \right\|_2,$$

and where Anderson requires that $\beta_n > 0$. Here Anderson considers a depth of M difference-vectors.

¹ Anderson uses θ_i^n to denote the i^{th} update coefficient of the n^{th} iterate, where we have written $\tilde{z}_i^{(n)}$ in keeping with the presentation of NKA.

The problem of finding a root of f may be recast as a fixed point problem by defining $G(\mathbf{x}) = \mathbf{x} - f(\mathbf{x})$, and then $\mathbf{r}_i = -f(\mathbf{x}_i)$. If, for each n , one defines the vectors

$$\tilde{\mathbf{v}}_i^{(n)} = \mathbf{x}_n - \mathbf{x}_{n-i} \quad \text{and} \quad \tilde{\mathbf{w}}_i^{(n)} = f(\mathbf{x}_n) - f(\mathbf{x}_{n-i}) \quad \text{for } i = 1, \dots, M,$$

then the Anderson Mixing update becomes

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left[\sum_{i=1}^M \tilde{z}_i^{(n)} \tilde{\mathbf{v}}_i^{(n)} + \beta_n \left(f(\mathbf{x}_n) - \sum_{i=1}^M \tilde{z}_i^{(n)} \tilde{\mathbf{w}}_i^{(n)} \right) \right],$$

where

$$\tilde{\mathbf{z}}^{(n)} = \arg \min_{\mathbf{y} \in \mathbb{R}^M} \left\| f(\mathbf{x}_n) - \sum_{i=1}^M y_i \tilde{\mathbf{w}}_i^{(n)} \right\|_2.$$

Note that $\text{span}\{\tilde{\mathbf{w}}_1^{(n)}, \dots, \tilde{\mathbf{w}}_M^{(n)}\} = \text{span}\{\mathbf{w}_{n-M+1}, \dots, \mathbf{w}_n\} = \mathcal{W}_n$. In particular the update coefficients of NKA and Anderson Mixing are related by the formula

$$\tilde{z}_i^{(n)} = - \sum_{j=n-i+1}^M \tilde{z}_j^{(n)}. \quad (3)$$

Further, for a constant and invertible derivative, $Df^{-1} \tilde{\mathbf{w}}_i^{(n)} = \tilde{\mathbf{v}}_i^{(n)}$. With this it is clear that, for linear problems, NKA is equivalent to Anderson Mixing when $\beta_n = 1$; the difference is only in the choice of basis vectors that span \mathcal{W}_n resulting in a change of variables given by Eq. (3). One advantage of the NKA formulation is that it uses differences of successive function evaluations to form a basis for \mathcal{W}_n . These differences can be used for M successive iterations. In contrast, Anderson's formulation forms a basis for \mathcal{W}_n using differences of the most recent function evaluation with all previous evaluations, which must be recomputed at every iteration.

2.3. Analytic Examinations of Anderson Mixing Applied to a Linear Problem

In [14], Walker and Ni consider the behavior of a generalized Anderson Mixing algorithm when it is applied to linear problems and when all previous vectors are stored, i.e. when $M = n$. They prove that, under modest assumptions, a class of variants of Anderson Mixing are equivalent to GMRES. They consider Anderson Mixing as a means to find a fixed-point of G . Like Anderson they compute $\mathbf{r}_i = G(\mathbf{x}_i) - \mathbf{x}_i$ at each step.

In their presentation of Anderson Mixing, though, they compute the update coefficients².

$$\tilde{\mathbf{z}}^{(n)} = \arg \min_{\mathbf{y} \in \mathbb{R}^{n+1}} \left\| \sum_{i=0}^n y_i \mathbf{r}_i \right\|_2 \quad \text{subject to} \quad \sum_{i=0}^n \tilde{z}_i^{(n)} = 1,$$

which are then used to form the update

$$\mathbf{x}_{n+1} = \sum_{i=0}^n \tilde{z}_i^{(n)} G(\mathbf{x}_i).$$

Again choosing $G(\mathbf{x}) = \mathbf{x} - f(\mathbf{x})$ (and hence $\mathbf{r}_i = -f(\mathbf{x}_i)$) and noting that the constraint requires that

$$\tilde{z}_n^{(n)} = 1 - \sum_{i=0}^{n-1} \tilde{z}_i^{(n)},$$

²Walker and Ni used f_j and $\alpha_n^{(j)}$ where we, for consistency, are using \mathbf{r}_j and $\tilde{z}_j^{(n)}$.

one has, as noted by Walker and Ni, that the above is equivalent to Anderson's original formulation

$$\tilde{\mathbf{z}}^{(n)} = \arg \min_{\mathbf{y} \in \mathbb{R}^n} \left\| f(\mathbf{x}_n) - \sum_{i=0}^{n-1} y_i (f(\mathbf{x}_n) - f(\mathbf{x}_i)) \right\|_2$$

and

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left[\sum_{i=0}^{n-1} \tilde{z}_i^{(n)} (\mathbf{x}_n - \mathbf{x}_i) + \left(f(\mathbf{x}_n) - \sum_{i=0}^{n-1} \tilde{z}_i^{(n)} (f(\mathbf{x}_n) - f(\mathbf{x}_i)) \right) \right].$$

Because Anderson Mixing and NKA are equivalent to each other and to the Walker and Ni formulation, we may restate one of Walker and Ni's theorems as follows:

Theorem 2.1 (Walker and Ni [14]). *Suppose that we search for a root of the function $f(\mathbf{x}) = \mathbf{Ax} - \mathbf{b}$ starting with \mathbf{x}_0 using either the NKA update or Anderson's update with $\beta_n = 1$. Suppose further that \mathbf{A} is non-singular. Let $\mathbf{x}_n^{\text{GMRES}}$ denote the n^{th} iterate generated by GMRES applied to the problem $\mathbf{Ax} = \mathbf{b}$ with starting point \mathbf{x}_0 and let $\mathbf{r}_n^{\text{GMRES}} = \mathbf{Ax}_n^{\text{GMRES}} - \mathbf{b}$ denote the associated residual. If for some $n > 0$, $\mathbf{r}_{n-1}^{\text{GMRES}} \neq 0$ and $\|\mathbf{r}_j^{\text{GMRES}}\|_2 < \|\mathbf{r}_{j-1}^{\text{GMRES}}\|_2$ for all $0 < j < n$, then the $n+1$ iterate generated by either update rule is given by $\mathbf{x}_{n+1} = (\mathbf{I} - \mathbf{A})\mathbf{x}_n^{\text{GMRES}} + \mathbf{b}$.*

It should be noted that the original presentation of the theorem given in [14] applies to a class of orthogonal projection methods where we have presented the theorem as applied to two members of this class: Anderson's original method and the NKA variant. It should also be noted that this theorem of Walker and Ni shows that

$$\mathcal{W}_n = \mathbf{AK}_n,$$

where $\mathcal{K}_n = \text{span}\{\mathbf{r}_0, \mathbf{Ar}_0, \dots, \mathbf{A}^{n-1}\mathbf{r}_0\}$.

An immediate corollary is

Corollary 2.2. *Let $\mathbf{r}_n = \mathbf{Ax}_n - \mathbf{b}$ denote the residual associated with the n^{th} iterate generated by either Anderson Mixing or NKA. Under the assumptions of Theorem 2.1*

$$\|\mathbf{r}_{n+1}\|_2 \leq \|\mathbf{I} - \mathbf{A}\| \|\mathbf{r}_n^{\text{GMRES}}\|_2,$$

where $\|\cdot\|$ is the L^2 operator norm.

Proof.

$$\begin{aligned} \mathbf{r}_{n+1} &= \mathbf{Ax}_{n+1} - \mathbf{b} \\ &= \mathbf{A} \left[(\mathbf{I} - \mathbf{A})\mathbf{x}_n^{\text{GMRES}} + \mathbf{b} \right] - \mathbf{b} \\ &= \mathbf{Ax}_n^{\text{GMRES}} - \mathbf{b} - \mathbf{A}(\mathbf{Ax}_n^{\text{GMRES}} - \mathbf{b}) \\ &= (\mathbf{I} - \mathbf{A})(\mathbf{Ax}_n^{\text{GMRES}} - \mathbf{b}), \end{aligned}$$

from which the claim follows. \square

The value of Corollary 2.2 is that, in the linear case, convergence of the non-truncated versions of both NKA and Anderson Mixing have the same characterizations as GMRES, i.e. when considering a normal matrix \mathbf{A} , the residual is controlled by the spectrum of \mathbf{A} , provided as GMRES doesn't stagnate.

2.4. Non-stagnation of GMRES

When considering a linear problem, the coefficients $\tilde{\mathbf{z}}^{(n)}$ (Anderson Mixing) and $\mathbf{z}^{(n)}$ (NKA) do two things. Through a minimization process, they select the best approximation within a given subspace, and they also select the subspace for the next iteration. The failure mode is that the best approximation within a given subspace doesn't use the information from the most recent iteration, in which case the next subspace will be the same as the current one and Anderson Mixing and NKA will become trapped. Anderson briefly discusses this in his presentation of the

method³. Lemma 2.1.5 from Ni’s dissertation [17] addresses this more directly. What Walker and Ni recognize in [14] is that this failure mode corresponds to the stagnation of GMRES.

There have been several examinations of when GMRES stagnates. Zavorian, O’Leary and Elman in [18] and Greenbaum, Ptak and Strakos in [19] present examples where GMRES does not decrease the norm of the residual on several successive iterations, i.e. it stagnates. While GMRES converges for such cases, Anderson Mixing and NKA will not. Greenbaum and Strakos show in [20] that the residual for GMRES applied to a matrix \mathbf{A} is strictly decreasing, i.e. no stagnation, if $\langle \mathbf{r}, \mathbf{A}\mathbf{r} \rangle \neq 0$ for all \mathbf{r} satisfying $\|\mathbf{r}\|_2 \neq 0$. This in turn will ensure the convergence of Anderson Mixing and NKA.

It is important to bear in mind that, in practice, the subspace \mathcal{W}_n is likely to have modest dimension, say 10, and is sitting in \mathbb{R}^N where N is much larger, often on the order of thousands or millions. The slightest perturbation of a vector will, with probability one, push it off of this low dimensional space. Even so, there is a simple change to the update step that provides a theoretical guarantee that NKA will still converge even when GMRES stagnates. This guarantee comes at a slight performance cost and is accomplished by modifying the update coefficients to ensure that $\mathcal{W}_{n-1} \subsetneq \mathcal{W}_n$.

We consider a modification to the NKA update rule⁴:

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left[\sum_{i=1}^n \mathbf{v}_i z_i^{(n)} + \left(f(\mathbf{x}_n) - \sum_{i=1}^n \mathbf{w}_i z_i^{(n)} \right) \right],$$

where $\mathbf{z}^{(n)}$ is chosen to minimize

$$\left\| f(\mathbf{x}_n) - \sum_{i=1}^n \mathbf{w}_i z_i^{(n)} \right\|_2.$$

We now add the following safety check: if $\|\mathbf{w}_n\|_2 \neq 0$, then perform the following update

$$z_n^{(n)} \leftarrow z_n^{(n)} \pm \varepsilon \frac{\left\| f(\mathbf{x}_n) - \sum_{i=1}^n \mathbf{w}_i z_i^{(n)} \right\|_2}{\|\mathbf{w}_n\|_2}$$

for some positive ε , where one chooses to add or subtract based on which will make the modified version of $|z_n^{(n)}| + 1$ further from zero. As will be shown in the proof of Theorem 2.3 the numerator of the modification is zero only when NKA has found the root.

With this we can strengthen Corollary 2.2 as it applies to NKA as follows:

Theorem 2.3. *Let \mathbf{A} be non-singular square matrix and suppose that we search for a root of the function $f(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$ starting with \mathbf{x}_0 using the modified version of NKA. Let $\mathbf{r}_n^{GMRES} = \mathbf{A}\mathbf{x}_n^{GMRES} - \mathbf{b}$ denote the n^{th} GMRES residual and let \mathbf{r}_n denote the n^{th} NKA residual, then*

$$\|\mathbf{r}_{n+1}\|_2 \leq (1 + \varepsilon) \|\mathbf{I} - \mathbf{A}\| \|\mathbf{r}_n^{GMRES}\|_2.$$

The proof can be found in Appendix A. The limitation of this result is that the orthogonal projection will become more poorly conditioned as ε decreases. Note that for the same reason both Anderson Mixing and NKA can develop arbitrarily poorly conditioned projection problems.

2.5. Relationship Between Anderson Mixing and Broyden

Fang and Saad in [21] provide a comparison between several methods including Anderson Mixing and Broyden. As noted above, the central difference between the two is that Broyden uses secant information to compute low rank updates to the approximation of Df , where Anderson Mixing uses the same information to compute arbitrary rank updates to Df^{-1} . Fang and Saad clarify this relationship and place a variant of Broyden and Anderson Mixing in a broader class of multi-secant update methods.

³This need to expand the space over which one is searching necessitates $\beta_n \neq 0$.

⁴The following could be adapted to many formulations of Anderson Mixing

Fang's and Saad's presentation of Anderson Mixing (cf. Eq. (24) in [21]) in the notation adopted in this paper is

$$\mathbf{x}_{n+1} = \mathbf{x}_n - \left[\sum_{i=n-M+1}^n \mathbf{v}_i z_i^{(n)} - \tilde{\beta} \left(f(\mathbf{x}_n) - \sum_{i=n-M+1}^n \mathbf{w}_i z_i^{(n)} \right) \right],$$

where

$$\mathbf{z}^{(n)} = \arg \min_{\mathbf{y} \in \mathbb{R}^M} \left\| f(\mathbf{x}_n) - \sum_{i=n-M+1}^n \mathbf{w}_i y_i \right\|_2,$$

and where $\tilde{\beta}$ is chosen to be positive.

It is worth noting that Fang and Saad have presented a version of Anderson Mixing that is more closely related to Carlson's and Miller's NKA method, but where NKA would require that, in the above formulation, $\tilde{\beta} = -1$. This leads to the question what is the best value of $\tilde{\beta}$ for a given problem, or in the case of Anderson's presentation what is the best choice for $\beta_n = -\tilde{\beta}$. We report numerical results for $\tilde{\beta} = -1$ (the NKA formulation) and $\tilde{\beta} = 1$ (denoted NKA_{-1}).

2.6. NKA in Practice

For our experiments we shall use the NKA formulation of Anderson Mixing. We shall let $\tilde{\beta} = 1$ and $\tilde{\beta} = -1$, as defined by Fang and Saad. Each \mathbf{w}_i vector is rescaled to have unit-norm, and the associated \mathbf{v}_i vector is scaled by the same factor. In our implementation we use a Cholesky Factorization to solve the least-squares problem associated with the orthogonal projection. This requires that the Gramian of the \mathbf{w}_i vectors be positive definite. We enforce this by choosing a linearly independent subset of the \mathbf{w}_i vectors as follows: At the beginning of iteration n we start with \mathbf{w}_n and consider \mathbf{w}_{n-1} . If the angle between \mathbf{w}_n and \mathbf{w}_{n-1} is less than some tolerance we discard \mathbf{w}_{n-1} . We iterate in this manner over successively older vectors, keeping them if the angle they make with the space spanned by the kept \mathbf{w}_i vectors is greater than the tolerance. This ensures that we may use Cholesky factorization and that the condition number of the problem is bounded.

Memory constraints require that $M < n$, forcing us to use NKA, NKA_{-1} and Broyden in a setting for which there are no theoretical results that the authors of this paper know of. One important distinction between Broyden and NKA is that for each iteration NKA stores a pair of state vectors, while Broyden only stores one. Consequently the memory requirements for Broyden are half that of NKA for the same depth of stored vectors.

Depending on the tolerance chosen at each linear solve for JFNK, one can reasonably expect theoretical guarantees of quadratic convergence to hold in some neighborhood of the solution, however identifying that neighborhood is often considerably harder than solving the original problem. In summary, for the numerical experiments we present, there is little theory regarding performance, making the following numerical results of value.

3. The k-Eigenvalue Formulation of the Boltzmann Transport Equation

The problem that we use to compare the efficiency of these various nonlinear methods is the k-eigenvalue formulation of the linearized Boltzmann neutron transport equation.

$$\left[\hat{\Omega} \cdot \vec{\nabla} + \Sigma_t(\vec{r}, E) \right] \psi(\vec{r}, \hat{\Omega}, E) = \int dE' \int d\Omega' \Sigma_s(\vec{r}, E' \rightarrow E, \hat{\Omega}' \cdot \hat{\Omega}) \psi(\vec{r}, \hat{\Omega}', E') + \frac{1}{k} \int dE' \chi(E' \rightarrow E) \bar{\nu} \Sigma_f(\vec{r}, E') \phi(\vec{r}, E'). \quad (4)$$

The unknown ψ describes the neutron flux in space, $\vec{r} \in \mathbb{R}^3$, angle $\hat{\Omega} \in \mathbb{S}^2$ and energy E . Σ_t is the total cross section, Σ_s is the scattering cross section and Σ_f is the fission cross section. The quantities $\bar{\nu}$ and χ characterize the rate and energy spectrum of neutrons emitted in the fission process. Integrating the flux over angle gives ϕ , the scalar flux at a given position and energy. For a thorough examination of the mathematical models of fission, including the Boltzmann transport equation, cf. [1, 2].

Discretization of Eq. (4) is accomplished using

1. S_N (discrete ordinates) in angle: An S_4 level symmetric quadrature set is chosen and the solution is computed at the abscissas of that set and then integrated over angle using the quadrature weights.
2. Multigroup in energy: Cross sections can be very noisy and it is therefore not practical to discretize the energy variable as one would a continuous function. Instead, the energy space is divided up into groups and it is assumed that the energy component for a cross section σ can be separated out:

$$\begin{aligned}\sigma(\vec{r}, E) &\approx f(E)\sigma_g(\vec{r}), & E_g < E \leq E_{g-1} \\ \sigma_g &= \frac{\int_{E_g}^{E_{g-1}} dE \sigma(\vec{r}, E)}{\int_{E_g}^{E_{g-1}} dE f(E)}\end{aligned}$$

3. Finite element or difference in space: The spatial variable can be treated in a variety of different ways. Here we explore results from two different discretization strategies as employed by the Los Alamos National Laboratory production transport codes PARTISN [22] and Capsaicin. PARTISN is used to generate results on a structured mesh with diamond (central) difference and Capsaicin to generate results on an unstructured polygonal mesh using discontinuous finite elements.

Applying the S_N and multigroup approximations, the linearized Boltzmann equation takes the form

$$\left(\hat{\Omega}_m \cdot \nabla + \sigma_{t,g}(\vec{r})\right)\psi_{g,m}(\vec{r}) = \frac{1}{4\pi} \sum_{g'=1}^G \sigma_{s,g' \rightarrow g}(\vec{r})\phi_{g'}(\vec{r}) + \frac{1}{k} \sum_{g'=1}^G \bar{\nu} \sigma_{f,g'}(\vec{r}) \frac{\chi_{g' \rightarrow g}(\vec{r})}{4\pi} \phi_{g'}(\vec{r}). \quad (5)$$

Here,

- $\psi_{g,m}$ represents the angular flux in direction $\hat{\Omega}_m$ in energy group g , which is the number of neutrons passing through some unit area per unit time $\left(\frac{\#}{\text{cm}^2 \cdot \text{Mev} \cdot \text{ster} \cdot \text{sec}}\right)$
- ϕ_g is the scalar flux, or angle-integrated angular flux. The S_N quadrature used to define angular abscissas ($\hat{\Omega}_m$) can be used to integrate ψ over all angle: $\phi_g = \sum_{m=1}^M w_m \psi_{g,m}$ where w_m are the quadrature weights $\left(\frac{\#}{\text{cm}^2 \cdot \text{Mev} \cdot \text{sec}}\right)$
- $\sigma_{t,g}$ is the total cross section, or interaction probability per area, for group g $\left(\frac{\#}{\text{cm}^2}\right)$
- $\sigma_{s,g' \rightarrow g}$ is the ‘inscatter’ cross section, which is the probability per area that a neutron will scatter from group g' into group g $\left(\frac{\#}{\text{cm}^2}\right)$
- $\sigma_{f,g}$ is the fission cross section for group g $\left(\frac{\#}{\text{cm}^2}\right)$
- $\bar{\nu}$ is the mean number of neutrons produced per fission event
- $\chi_{g' \rightarrow g}$ describes the energy spectrum of emitted fission neutrons in group g produced by neutrons absorbed from group g'

Application of the spatial discretization then yields a matrix equation. For convenience, this equation can be expressed in operator notation as

$$\mathbf{L}\psi = \mathbf{MSD}\psi + \frac{1}{k}\mathbf{MFD}\psi. \quad (6)$$

where \mathbf{L} is the streaming and removal operator, \mathbf{S} is the scattering operator, \mathbf{F} is the fission operator, and \mathbf{M} and \mathbf{D} are the moment-to-discrete and discrete-to-moment operators, respectively, and $\mathbf{D}\psi = \phi$.

If the fluxes are arranged by energy group, the form of \mathbf{L} is block diagonal and it can be inverted onto a constant right-hand-side exactly by performing a so-called sweep for each energy group and angular abscissa. The sweep is an exact application of \mathbf{L}^{-1} and can be thought of as tracking the movement of particles through the mesh along a single direction beginning at the incident boundaries, which vary by angle, so that all necessary information about neutrons streaming into a cell from other ‘upwind’ cells is known before it is necessary to make calculations for that cell.

Most transport codes contain the mechanism to perform the sweep, but could not apply \mathbf{L} directly without significant modification, therefore Eq. (6) is generally thought of as a standard eigenproblem of the form

$$k\phi = (\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS})^{-1} \mathbf{DL}^{-1}\mathbf{MF}\phi.$$

Because we seek the mode with the largest magnitude eigenvalue, power iteration is one possible iterative technique:

$$\begin{aligned}\phi_{z+1} &= (\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS})^{-1} \mathbf{DL}^{-1}\mathbf{MF}\phi_z \\ k_{z+1} &= \frac{W^T \mathbf{F}\phi_{z+1}}{W^T \mathbf{F}\phi_z}\end{aligned}$$

Full inversion of $(\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS})$ is expensive, however, so it is generally more efficient to use a nonlinear fixed point iteration (FPI) in which the operator $(\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS})$ is inverted only approximately using a nested inner-outer iteration scheme to converge the scattering term. Commonly, one or more ‘outer iterations’ are conducted in which the fission source is lagged. Each outer iteration consists of a series of inner ‘within-group’ iterations (one or more per energy group) in which the inscatter is lagged so that the within-group scattering can be converged. In general, the iteration can be written as

$$\phi_{z+1} = \mathbf{P}(k_z)\phi_z \tag{8a}$$

$$k_{z+1} = k_z \frac{W^T \mathbf{F}\phi_{z+1}}{W^T \mathbf{F}\phi_z} \tag{8b}$$

where W is a vector of weights and generally represents a volume integral. The k -eigenvalue can be thought of as the ratio of neutrons in successive generations and new neutrons are created in fission events, therefore it makes sense to adjust k using the ratio of fission sources. If $\mathbf{P}(k_z) = (\mathbf{I} - \mathbf{DL}^{-1}\mathbf{MS})^{-1} \mathbf{DL}^{-1}\mathbf{MF}$ we recover a true power iteration, but there are numerous possible forms for this operator. It is typically more efficient to limit the number of sweeps, however. The most basic approach is to apply \mathbf{L}^{-1} via a sweep. In light of the fact that we are lagging the scattering altogether, we also consider another update for the eigenvalue that accounts for the change in the scattering as well.

$$\phi_{z+1} = \mathbf{DL}^{-1}\mathbf{M}\left(\mathbf{S} + \frac{1}{k_z}\mathbf{F}\right)\phi_z \tag{9a}$$

$$k_{z+1} = \frac{W^T \mathbf{F}\phi_{z+1}}{W^T \left(\frac{1}{k_z}\mathbf{F}\phi_z - \mathbf{S}(\phi_{z+1} - \phi_z)\right)}. \tag{9b}$$

Clearly, if we assume that the scattering is converged, then the second term in the denominator of Eq. (9b) is zero and we recover Eq. (8b). For a converged system, this term will indeed be zero and the ratio of the fission sources will go to one because the fission source has been suitably adjusted by $\frac{1}{k}$ so that the net neutron production is zero.

From this we have that a fixed point of the iteration in Eq. (9) is also a root of the residual function

$$F\left(\begin{array}{c} \phi \\ k \end{array}\right) = \left(\begin{array}{c} \mathbf{P}(k)\phi \\ 1 - \frac{W^T \mathbf{F}\phi}{W^T \mathbf{F}\phi}k \end{array}\right), \tag{10}$$

where

$$\mathbf{P}(k) = (\mathbf{I} - \mathbf{DL}^{-1}\mathbf{M}\left(\mathbf{S} + \frac{1}{k}\mathbf{F}\right))$$

In order to initialize the iteration, a single sweep is performed on a vector of ones, $E = [1 \ 1 \ \dots \ 1]^T$, and the scaled two-norm of the flux is then normalized to one:

$$\begin{aligned}\phi_0 &= \frac{\mathbf{DL}^{-1}\mathbf{M}(\mathbf{S} + \mathbf{F}) E}{\|\mathbf{DL}^{-1}\mathbf{M}(\mathbf{S} + \mathbf{F}) E\|_{2,s}} \\ k_0 &= 1\end{aligned}$$

Here $\|\cdot\|_{2,s}$ indicates the L^2 -norm normalized by the square root of the number of degrees of freedom. Once the residual is formulated, it is possible to apply any of the nonlinear solvers discussed to seek a root of F given in Eq. (10). The following section contains a comparison of JFNK, Broyden, NKA₋₁ and NKA for the k-eigenvalue problem.

4. Results

Results are given for a cylinder with a 3.5 cm radius and a height of 9 cm modeled in two-dimensional cylindrical coordinates, similar to the cylindrical problem that was studied in [4] in three-dimensional Cartesian coordinates. The problem consists of a central 5-cm layer of Boron-10 with 1 cm thick water layers on either side and 1 cm layers of highly enriched uranium on the ends. The top, bottom and radial boundaries are either all vacuum or all reflective (the inner radial boundary is a symmetry condition in cylindrical coordinates). This a ‘difficult’ problem (i.e., one with a high dominance ratio) because the problem is symmetric, having two fissile regions that are effectively ‘decoupled’ because of the reflector (water) and absorber (boron) between them, which means the second eigenmode is close to the fundamental mode; reflection further increases the difficulty of the problem. A 16-group Hansen-Roach cross section data set is used to generate the results presented here (the 16 group data was collapsed to 5 groups in the original paper [4]).

A 175 (r-axis) by 450 (z-axis) mesh of equally sized squares is used in PARTISN for both the reflected and unreflected problems. The Capsaicin results are computed on an unstructured mesh comprising roughly the same number of cells in the r and z axes as the PARTISN computation, for a total of 79855 (possibly non-convex) polygons, each of which has 3 to 6 vertexes on a cell. These codes use similar methods, but there are several differences that lead to different iterative behavior and slightly different results. Firstly, PARTISN is an orthogonal mesh code that utilizes a diamond (central) difference (DD) spatial discretization, requiring the storage of only one cell-centered value per energy group [2]. Capsaicin uses an unstructured mesh, which has the advantage of being able to model geometrically complex problems with higher fidelity, but it comes at a cost. Finding an efficient sweep schedule on an unstructured mesh that minimizes the latency in parallel computations is a difficult problem in itself [23, 24, 25]. In contrast, a parallel sweep schedule that is nearly optimal is easy to implement for structured meshes [26]. Furthermore, a discontinuous finite element method (DFEM) spatial discretization is employed in Capsaicin such that the number of unknowns per cell is equal to the number of nodes [27]. While the DFEM has better accuracy than DD, potentially enabling the use of commensurately fewer mesh cells for the same solution accuracy, it is more costly to compute the solution to the DFEM equations than the DD equations, and the DFEM is thus less efficient than DD when calculations are performed on meshes of similar size.

The second difference between the two codes is that reflecting boundary conditions can cause instabilities with the DD spatial discretization and the angular differencing used in PARTISN so, in order to achieve stability, the reflected flux is ‘relaxed’. This is done using a linear combination of the new and old reflected fluxes as the boundary source for the next iteration:

$$\psi_{relaxed}^{z+1} = r\psi_{refl}^z + (1 - r)\psi_{refl}^{z+1}.$$

The relaxation parameter, r , is, by default, $\frac{1}{2}$ for this problem. This relaxation is not necessary in Capsaicin because it uses a more accurate spatial and angular discretization, including the use of starting directions for calculations in cylindrical coordinates [2], as well the use of reflected starting directions when reflection conditions are specified on the radial boundary. The eigenvalues computed by PARTISN are $k = 0.1923165$ and $k = 0.8144675$ for the unreflected and reflected cases, respectively. Because the codes employ different discretizations, the eigenvalues calculated by Capsaicin were $k = 0.191714$ for the unreflected case and $k = 0.814461$ for the reflected case. The Capsaicin discretization results in roughly 50 million degrees of freedom while the PARTISN discretization results in a little over 5 million degrees of freedom.

The three nonlinear solvers that we consider in this paper are implemented in the NOX nonlinear solver package that is part of the larger software package Trilinos 10.6 [28]. We use JFNK as it is implemented in NOX and have written implementations of both NKA and Broyden’s method in the NOX framework as user supplied direction classes. These two direction classes are available for download on Sourceforge at <http://sourceforge.net/projects/>

nlkain/. In addition to the existing JFNK interface, interfaces to the NOX package were developed for the Broyden and NKA methods so that all methods except fixed-point iteration are accessed through the same Trilinos solver. JFNK can be extremely sensitive to the choice of forcing parameter, η , therefore we explored variations of the three possibilities implemented in NOX:

$$\text{Constant:} \quad \eta_z = \eta_0 \quad (12a)$$

$$\text{Type 1 (EW1):} \quad \eta_z = \left| \frac{\|F_z\| - \|J_{z-1}\delta_{z-1} + F_{z-1}\|}{\|F_{z-1}\|} \right|. \quad \text{If } \eta_{z-1}^{\frac{1+\sqrt{5}}{2}} > .1, \quad \text{then } \eta_z \leftarrow \max\left\{\eta_z, \eta_{z-1}^{\frac{1+\sqrt{5}}{2}}\right\}. \quad (12b)$$

$$\text{Type 2 (EW2):} \quad \eta_z = \gamma \left(\frac{\|F_z\|}{\|F_{z-1}\|} \right)^\alpha. \quad \text{If } \gamma \eta_{z-1}^\alpha > .1, \quad \text{then } \eta_z \leftarrow \max\{\eta_z, \gamma \eta_{z-1}^\alpha\}. \quad (12c)$$

Types 1 and 2 were developed by Eisenstat and Walker [29], therefore they are denoted EW1 and EW2 in the results that follow. The NOX default parameters were also used for EW1 and EW2: $\eta_0 = 10^{-1}$, $\eta_{\min} = 10^{-6}$, $\eta_{\max} = 10^{-2}$, $\alpha = 1.5$ and $\gamma = 0.9$. Note that for the results below, the convergence criterion is $\|F\|_{2,s} \leq 10^{-9}$.

4.1. Unreflected Cylinder

4.1.1. PARTISN

Tables 1a-1d show the number of JFNK and total inner GMRES iterations, the number of sweeps, the CPU time and the percentage of that time that was spent computing the residual, respectively. As can be seen in Tables 1a and 1b, the GMRES subspace size does not affect the number of Newton iterations for the most part and has little effect on the total number of GMRES iterations or sweeps. As the forcing parameter decreases, however, more sweeps are required because more GMRES iterations are required, both overall and per Newton step. Smaller subspace sizes require more restarts, which in turn each require one additional sweep. In general, for NKA, NKA with the $\beta = 1$ (NKA₋₁) and Broyden, decreasing the subspace size leads to an increase in sweep count, but the effect is negligible for NKA in this case, while it is significant for NKA₋₁ and Broyden. Broyden(10) also requires many more iterations than Broyden(5) in contradiction to the general trend. NKA₋₁ also failed to converge for a subspace size of 5. Overall, NKA requires the fewest sweeps and displays a predictable trend of decreasing sweep count with increasing subspace size. As Table 1c shows, the runtimes are consistent with the sweep count. NKA is more than 3 times faster than FPI and between 1.6 and 2.3 times as fast as JFNK, while NKA₋₁ is, at best, comparable to JFNK and, at worst, more than an order of magnitude slower than FPI. The behavior of Broyden is rather chaotic—at best, it is almost as efficient as NKA, but at worst it too is slower than FPI.

The percentage of the total time spent evaluating the residual is shown in Table 1d because, for this particular code and problem, the solver time requires a non-trivial portion of the run-time. As can be seen, this percentage is slightly smaller for NKA and NKA₋₁ than Broyden and noticeably smaller for Broyden than for JFNK. There is also a slight decrease as the subspace size increases in all cases, which is expected since the respective solvers must do more work for a larger subspace than a smaller one. JFNK requires the least amount of time in the solver with approximately 95% of the run-time spent in the sweep regardless of GMRES subspace size. Despite these numbers, we note that NKA still requires the least run-time of all of the methods.

The plots in Fig. 1 show the scaled L^2 -norm of the residual as a function of the number of sweeps. For JFNK where there are a few outer Newton iterations with multiple sweeps required to do the inner inversion of the Jacobian, the L^2 -norm is given at each Newton step plotted at the cumulative sweep count up to that point. Fig. 1a shows that the behavior of NKA is similar for all subspace sizes and much more efficient than FPI. Fig. 1b shows NKA₋₁, which converges fairly quickly for the larger restarts, but then has trouble reducing the residual for subspaces of 5 and 10. While 10 eventually converges, 5 seems to stagnate. Fig. 1c shows the behavior of Broyden. As can be seen, the norm fluctuates quite erratically for every subspace size except 30 and cannot compete with NKA with a subspace of 5. Fig. 1d shows some of the more efficient JFNK results plotted at each Newton iteration for the current sweep count. And finally, Fig. 1e shows a comparison of the most efficient results for each method.

Table 1: PARTISN unreflected cylinder: Number of outer and inner JFNK iterations, sweeps, run-time and percentage of the run-time spent computing the residual to an accuracy of $\|F\|_{2,s} \leq 10^{-9}$ for the various methods.

(a) Outer JFNK/total inner GMRES iterations

subspace	η				
	0.1	0.01	0.001	EW1	EW2
30	8 (30)	6 (42)	5 (45)	6 (39)	5 (38)
20	8 (30)	6 (42)	5 (45)	6 (39)	5 (38)
10	8 (30)	6 (42)	5 (48)	6 (41)	5 (42)
5	8 (30)	6 (48)	5 (50)	5 (43)	5 (44)

(b) Number of sweeps (FPI converged in 99)

subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	26	43	31	47	55	56	57	49
20	26	69	43	47	55	56	57	49
10	27	946	123	47	55	60	60	55
5	28	–	70	47	66	68	64	61

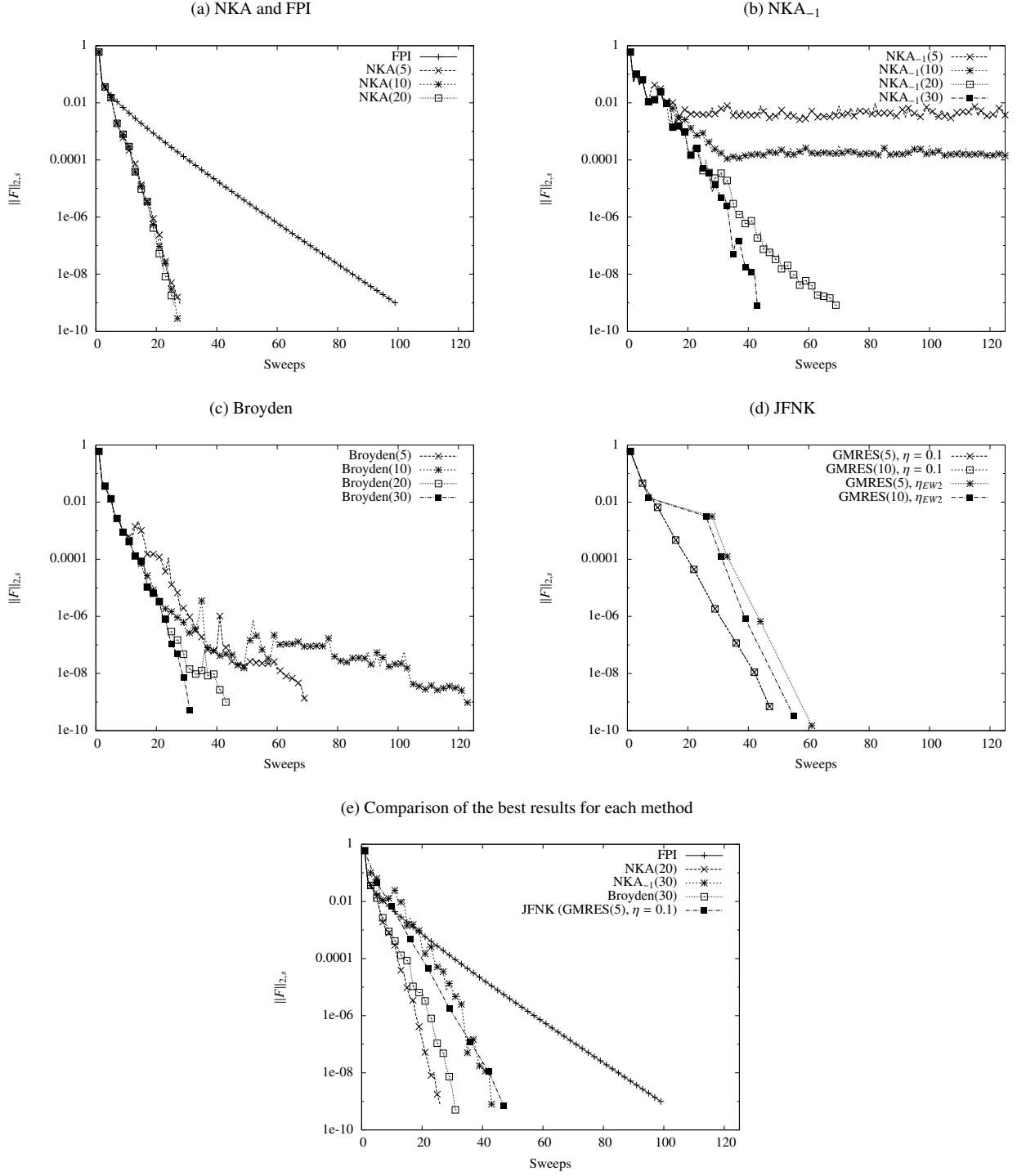
(c) CPU time (s) (FPI converged in 245.8 s)

subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	74.36	129.10	85.37	121.09	142.05	147.25	147.48	127.32
20	74.36	203.84	118.08	120.32	141.85	147.35	147.10	127.94
10	75.29	2657.78	328.29	121.40	141.84	156.61	154.53	141.82
5	75.81	–	182.20	121.03	169.41	174.05	164.40	158.09

(d) Percentage of CPU time spent in the residual evaluation

subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	85.79	81.56	89.67	95.70	95.02	94.75	94.93	94.65
20	86.30	82.78	89.47	95.71	95.03	94.76	94.93	94.66
10	89.05	87.39	92.05	95.74	95.02	94.88	95.20	95.12
5	91.77	–	94.36	95.74	95.70	95.75	95.80	95.79

Figure 1: PARTISN unreflected cylinder: Scaled L^2 -norm of the residual as a function of number of sweeps for the various methods and subspace sizes. Each of the methods is plotted on the same scale to simplify comparisons between panels. Note that, in panel (d), points plotted on the lines indicate when a JFNK iteration starts (JFNK requires multiple sweeps per iteration). In panels (a), (b), and (c) we plot one point per two iterations of the method. In panel (e) the convention for iterations per plotted points is the same as in panels (a) through (d).



4.1.2. Capsaicin results

The different spatial and angular discretization methods used in Capsaicin alter the numerical properties of the operators associated with the k-eigenvalue problem, affecting the convergence rates and curves of the various solution methods. While the methods in Capsaicin have better accuracy than those in PARTISN, they are also more costly due to the additional degrees of freedom associated with the DFEM spatial discretization and the use of starting directions in the angular discretization. This is evident in the relatively long run times associated with the Capsaicin calculations, even though they were computed with a parallel decomposition in energy, in addition to the parallel mesh decomposition, on 24 processors. Because of other differences in implementation between PARTISN and Capsaicin, a minimum of 99% of the computation time is spent in the evaluation of the residual (for all solution methods) and therefore is not reported in the tables.

Tables 2a-2c show the number of JFNK and total inner GMRES iterations, the number of sweeps and the CPU time that was spent computing the residual, respectively. As can be seen in Tables 2a and 2b, as for PARTISN the GMRES subspace size does not affect the number of Newton iterations for the most part and as the forcing parameter decreases more sweeps are required. However, the subspace size does affect the total number of GMRES iterations, hence the sweep count, for the smaller and varying forcing parameters as was not seen with PARTISN. Once again, for NKA, NKA_{-1} and Broyden, decreasing the subspace size leads to an overall increase in sweep count and Broyden(10) requires many more iterations than Broyden(5) in contradiction to the general trend. NKA_{-1} failed to converge for subspace sizes of 5 and 10 where PARTISN only failed for 5. Once again, NKA requires the fewest sweeps and displays a predictable trend of decreasing sweep count with increasing subspace size. Table 2c demonstrates that once again the runtimes are consistent with the sweep count. NKA is more than 6 times faster than FPI and between 1.8 and 3.6 times as fast as JFNK for comparable subspace sizes, while NKA_{-1} is comparable to JFNK *when it converges*. The behavior of Broyden is, once again, chaotic, at times almost as efficient as NKA, but at worst slower than JFNK, although in this case it is always more efficient than FPI.

The plots in Fig. 2 show the scaled L^2 -norm of the residual as a function of the number of sweeps. The behavior is similar to that seen in Fig. 1, FPI experiences an abrupt change in slope between $\|F\|_{2,s} = 10^{-8}$ and 10^{-9} that is not seen in the PARTISN results and leads to a much larger iteration count. It is also interesting to note that in the NKA_{-1} results, shown in Fig. 2b, subspaces of 5 and 10 seem to stagnate for large numbers of iterations instead of consistently dropping to zero as they do for NKA in Fig. 2a.

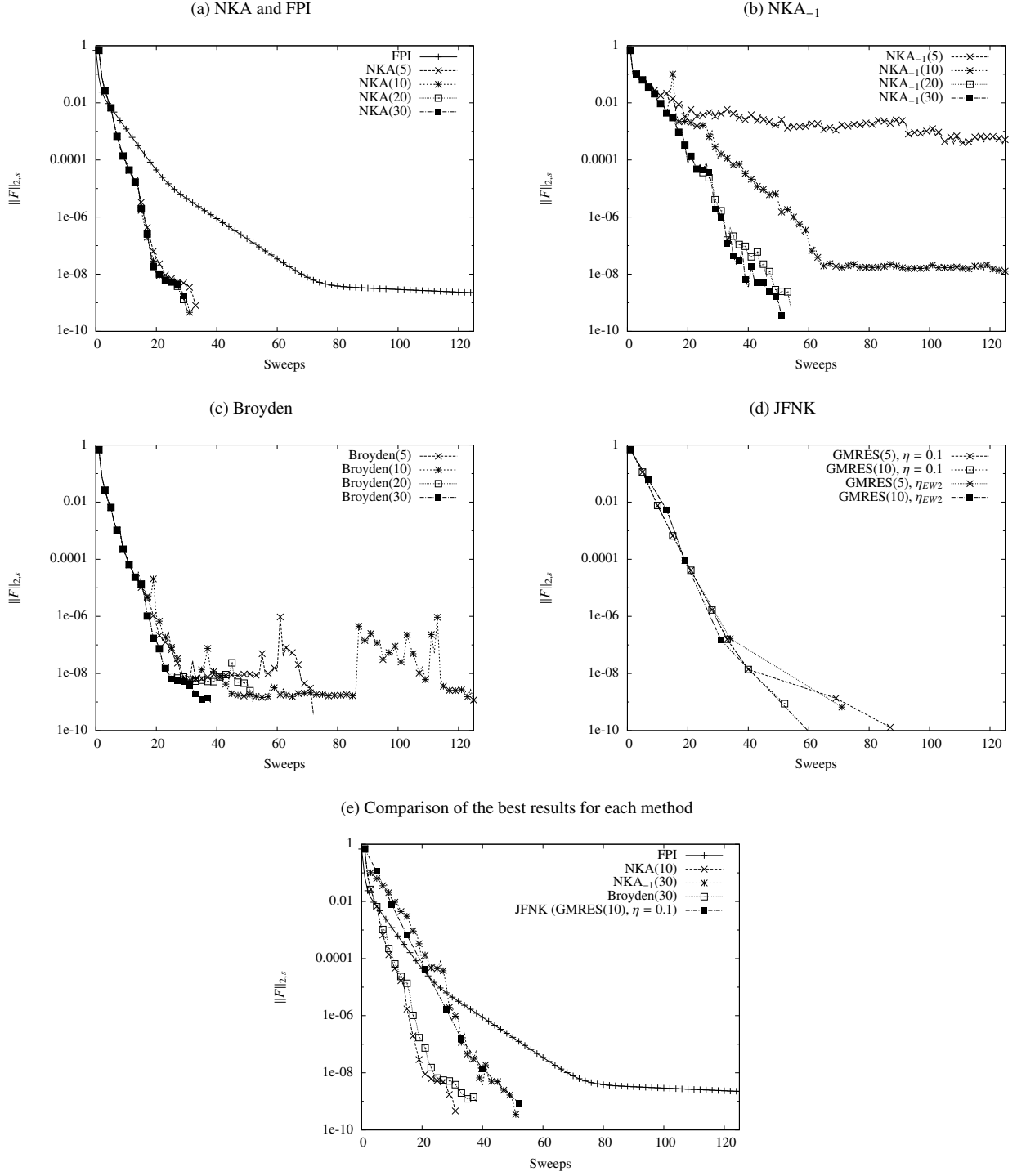
Table 2: Capsaicin unreflected cylinder: Number of outer and inner JFNK iterations, sweeps and run-time spent computing the residual to an accuracy of $\|F\|_{2,s} \leq 10^{-9}$ for the various methods.

(a) Outer JFNK/total inner GMRES iterations								
subspace	η							
	0.1	0.01	0.001	EW1	EW2			
30	8 (35)	6 (47)	5 (49)	5 (37)	5 (39)			
20	8 (35)	6 (47)	5 (49)	5 (37)	5 (39)			
10	8 (35)	6 (60)	5 (62)	5 (51)	5 (52)			
5	9 (62)	6 (66)	5 (68)	6 (85)	5 (53)			

(b) Number of sweeps (FPI converged in 202)								
subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	31	52	39	53	61	61	53	51
20	31	55	53	53	61	61	53	51
10	32	–	127	53	76	76	70	66
5	34	–	73	88	89	90	116	72

(c) CPU time (ks) (FPI converged in 12.71 ks)								
subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	1.9	3.3	2.4	3.4	4.2	3.9	3.4	3.4
20	1.9	3.6	3.4	3.4	4.1	4.0	3.5	3.3
10	2.1	–	8.2	3.5	5.0	5.1	4.7	4.3
5	2.1	–	4.5	5.7	5.9	5.9	7.6	4.7

Figure 2: Capsaicin unreflected cylinder: Scaled L^2 -norm of the residual as a function of number of sweeps for the various methods and subspace sizes. Each of the methods is plotted on the same scale to simplify comparisons between panels. Note that, in panel (d), points plotted on the lines indicate when a JFNK iteration starts (JFNK requires multiple sweeps per iteration). In panels (a), (b), and (c) we plot one point per two iterations of the method. In panel (e) the convention for iterations per plotted points is the same as in panels (a) through (d).



4.2. Fully Reflected Cylinder

4.2.1. PARTISN results

Here, we duplicate the results shown in 4.1.1, only this time, reflection is added to the top, bottom and outer edges of the cylinder. Tables 3a-3d show the number of JFNK and total inner GMRES iterations, the number of sweeps, the CPU time and the percentage of that time that was spent computing the residual, respectively. As can be seen in Tables 3a and 3b, increasing the subspace size has a non-negligible effect on Newton iteration count, and a significant effect on GMRES iteration count and sweep count. One additional parameter that comes into play for this problem is the maximum number of inner GMRES iterations allowed. This parameter was set to the NOX default of 30 for these computations and in many cases the inner iteration did not converge to the specified tolerance because it exceeded this limit. We note that, in our experience, increasing this upper bound tends to degrade the performance of JFNK, just as decreasing the forcing parameter often degrades the performance. For NKA and Broyden, decreasing the subspace size also leads to an increase in sweep/iteration count, quite noticeably in this case. The only subspace size that converged for NKA_{-1} was 30, which requires more sweeps than any other iterative method including FPI. For smaller subspace sizes, Broyden also fails to converge. When comparing the runtimes shown in Table 3c for all of the iterative methods, we see that JFNK with GMRES(5) is only slightly more efficient than FPI, while JFNK with GMRES(30) is comparable to NKA(5). Broyden, when it converges, is slower than NKA, but comparable to JFNK. The runtime for NKA_{-1} is an order of magnitude greater than any other method. Table 3d shows that the percentage of time spent in the residual is ranges from approximately 95% for JFNK with GMRES(5) to approximately 90% for GMRES(30). The percentage is smallest for NKA and NKA_{-1} , and slightly larger for Broyden. Once again, we note that despite the fact that a larger percentage of time is spent in the NKA solver than in JFNK or Broyden, it is still more efficient in terms of CPU time.

The plots in Fig. 3 show the scaled L^2 -norm of the residual as a function of the number of sweeps. Fig. 3a shows that the behavior of NKA is similar for all subspace sizes and much more efficient than FPI in all Fig. 3b shows that NKA_{-1} behaves erratically for subspaces of 5 and 10, and at 20 seems to stagnate before reaching the specified convergence criterion. Even when it does converge for a subspace size of 30, it is slower even than FPI. Fig. 3c shows the behavior of Broyden. As can be seen, the norm fluctuates erratically and, although it appears to be converging initially for subspaces of 5 and 10, it never achieves an error norm below 10^{-8} and eventually diverges. Broyden(30) and Broyden(20), which converge to the desired norm, cannot compete with NKA(20). Fig. 3d shows some of the more efficient JFNK results plotted at each Newton iteration for the current sweep count. And finally, Fig. 3e shows a comparison of the most efficient results for each method. Clearly, NKA(20) is significantly more efficient than the other iterative methods, although we note that NKA(10) is comparable to JFNK with GMRES(20).

Figure 3: PARTISN reflected cylinder: Scaled L^2 -norm of the residual as a function of number of sweeps for the various methods and subspace sizes. Each of the methods is plotted on the same scale to simplify comparisons between panels. Note that, in panel (d), points plotted on the lines indicate when a JFNK iteration starts (JFNK requires multiple sweeps per iteration). In panels (a), (b), and (c) we plot one point per six iterations of the method. In panel (e) the convention for iterations per plotted points is the same as in panels (a) through (d).

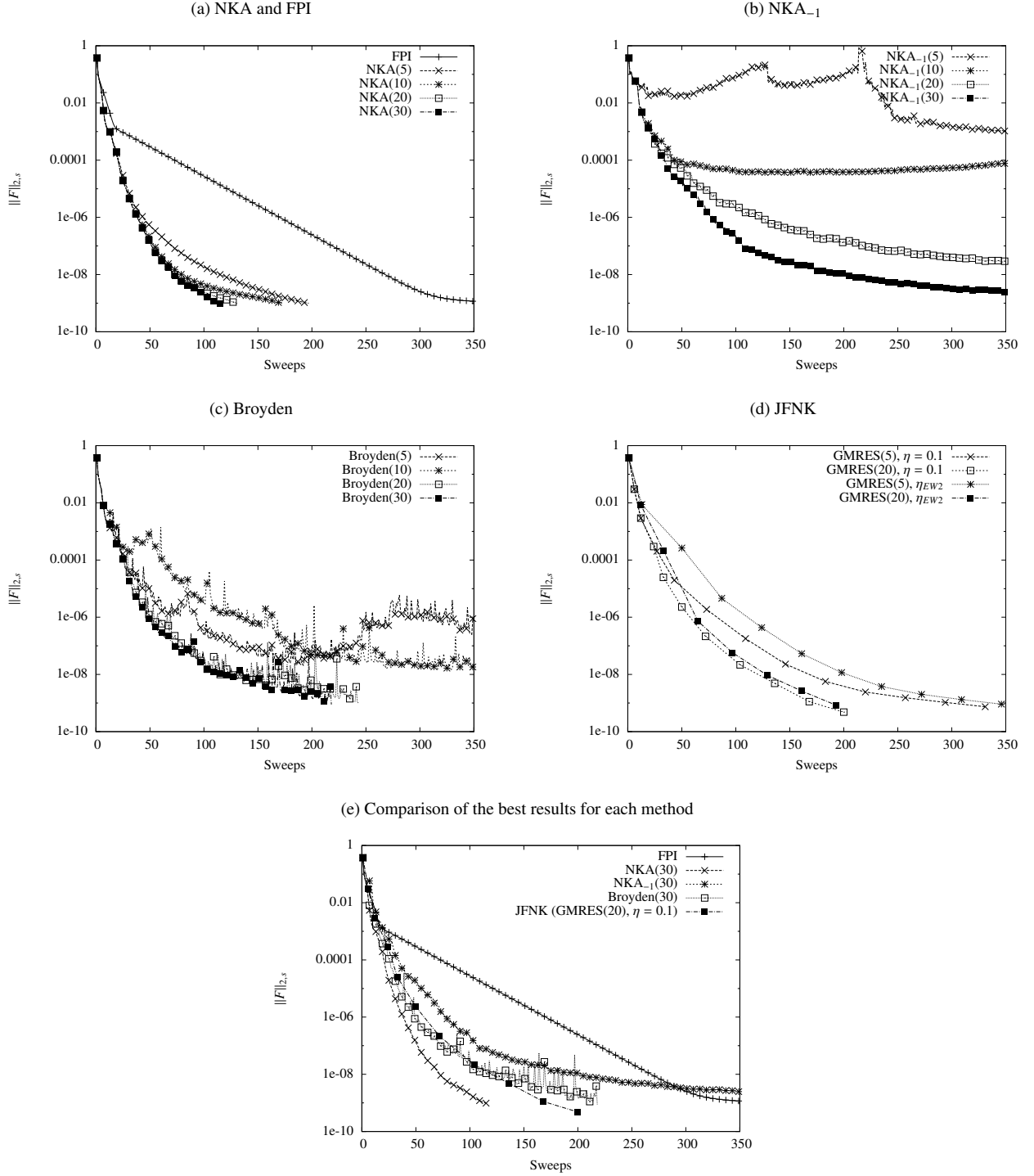


Table 3: PARTISN reflected cylinder: Number of outer and inner JFNK iterations, sweeps, run-time and percentage of the run-time spent computing the residual to an accuracy of $\|F\|_{2,s} \leq 10^{-9}$ for the various methods.

(a) Outer JFNK/total inner GMRES iterations								
subspace	η							
	0.1	0.01	0.001	EW1	EW2			
30	10 (177)	7 (167)	7 (192)	7 (165)	7 (175)			
20	10 (179)	8 (198)	7 (197)	8 (195)	7 (177)			
10	10 (186)	8 (202)	9 (257)	8 (196)	8 (212)			
5	12 (263)	10 (271)	10 (289)	10 (252)	10 (279)			

(b) Number of sweeps (FPI converged in 389)								
subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	115	594	218	198	182	207	186	190
20	131	–	243	200	216	212	220	193
10	173	–	–	218	232	293	233	243
5	197	–	–	331	337	358	323	346

(c) CPU time (s) (FPI converged in 971.7 s)								
subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	372.2	1975.1	645.7	536.8	498.6	569.8	509.8	523.3
20	398.0	–	686.6	529.7	577.9	565.4	591.2	516.0
10	486.3	–	–	572.1	613.4	769.9	613.2	639.9
5	535.7	–	–	855.8	895.0	932.4	842.2	906.0

(d) Percentage of CPU time spent in the residual evaluation								
subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	76.44	74.51	83.48	91.40	90.68	90.36	90.94	90.59
20	81.05	–	87.25	93.06	92.83	92.64	92.97	92.74
10	87.02	–	–	94.55	94.40	94.35	94.48	94.36
5	91.04	–	–	95.61	95.66	95.57	95.62	95.58

4.2.2. Capsaicin results

Here, we duplicate the results shown in 4.1.2, only this time, reflection is added to the top, bottom and outer edges of the cylinder. Tables 4a-4c show the number of JFNK and total inner GMRES iterations, the number of sweeps and the CPU time that was spent computing the residual, respectively. As can be seen in Tables 3a and 3b, increasing the subspace size does not affect the Newton iteration count, but it does have a significant effect on the GMRES iteration count and sweep count. In this regard, the behavior is more similar to the unreflected case than the PARTISN reflected case. For NKA and Broyden, there is an overall increase in iteration count, but for Broyden it is slight compared to that seen for the unreflected cylinder and the reflected PARTISN results. Also, for Capsaicin, Broyden always converges whereas for PARTISN only subspaces of 20 and 30 converged. The only subspace size that converged for NKA₋₁ was 30, which is consistent with the PARTISN results, and which once again requires more sweeps than any other iterative method including FPI. When comparing the runtimes shown in Table 4c for all of the iterative methods, we see that JFNK with GMRES(5) is generally slower than FPI, while JFNK with GMRES(30) is slower than NKA with a subspace of 5. Broyden is slower than NKA, but always more efficient than JFNK. The runtime for NKA₋₁ is twice that for FPI.

The plots in Fig. 3 show the scaled L^2 -norm of the residual as a function of the number of sweeps. Fig. 4a shows that the behavior of NKA is similar for all subspace sizes and much more efficient than FPI while Fig. 4b shows that NKA₋₁ seems to stagnate before reaching the specified convergence criterion subspaces of 5 and 10, and at 20. Even when it does converge for a subspace size of 30, it is slower even than FPI. Fig. 4c shows the behavior of Broyden. As can be seen, the convergence is fairly regular, in sharp contrast to the PARTISN results where the norm fluctuated erratically and often failed to converge. Fig. 4d shows some of the more efficient JFNK results plotted at each Newton iteration for the current sweep count. And finally, Fig. 3e shows a comparison of the most efficient results for each method. Clearly, NKA(20) is more efficient than the other iterative methods.

Figure 4: Capsaicin reflected cylinder: Scaled L^2 -norm of the residual as a function of number of sweeps for the various methods and subspace sizes. Each of the methods is plotted on the same scale to simplify comparisons between panels. Note that, in panel (d), points plotted on the lines indicate when a JFNK iteration starts (JFNK requires multiple sweeps per iteration). In panels (a), (b), and (c) we plot one point per two iterations of the method. In panel (e) the convention for iterations per plotted points is the same as in panels (a) through (d).

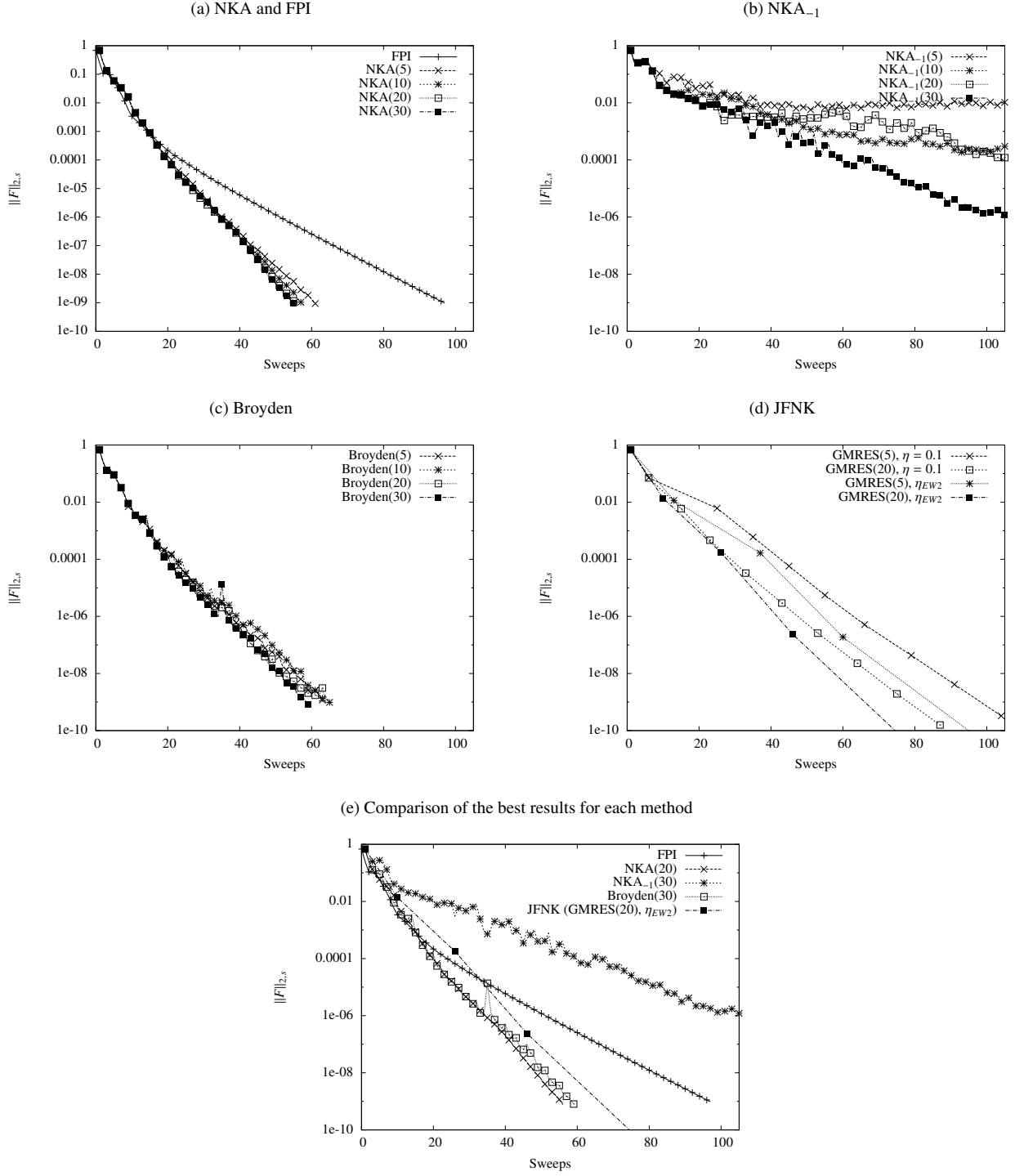


Table 4: Capsaicin reflected cylinder: Number of outer and inner JFNK iterations, sweeps and run-time spent computing the residual to an accuracy of $\|F\|_{2,s} \leq 10^{-9}$ for the various methods.

(a) Outer JFNK/total inner GMRES iterations								
subspace	η							
	0.1	0.01	0.001	EW1	EW2			
30	9 (68)	5 (64)	4 (78)	4 (61)	4 (69)			
20	9 (68)	5 (64)	4 (87)	5 (74)	4 (69)			
10	9 (68)	5 (68)	4 (91)	4 (62)	4 (72)			
5	9 (76)	5 (137)	4 (95)	4 (66)	4 (76)			

(b) Number of sweeps (FPI converged in 99)								
subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	56	202	60	88	76	88	74	79
20	57	–	65	88	76	98	90	79
10	56	–	66	88	84	107	79	86
5	62	–	65	105	95	121	89	98

(c) CPU time (ks) (FPI converged in 6.25 ks)								
subspace	NKA	NKA ₋₁	Broyden	JFNK η				
				0.1	0.01	0.001	EW1	EW2
30	3.5	12.8	3.9	5.9	5.2	5.7	5.0	5.2
20	3.7	–	4.0	5.8	5.0	6.4	6.0	5.2
10	3.6	–	4.2	5.7	5.3	6.9	5.2	5.6
5	3.9	–	4.2	6.7	6.5	7.8	5.8	6.6

5. Conclusion

The benefit of JFNK is its “Newton-like” convergence achieved by developing an arbitrarily accurate approximation of the inverse of the Jacobian at each ‘outer’ iteration, but the cost is repeated function evaluations in each of these ‘outer’ iterations and the wasteful discarding of potentially useful subspace information. In contrast Broyden and Anderson Mixing only perform a single function evaluation at each iteration, but continue to use ‘old’ information from previous iterations to improve their estimate of the Jacobian. The drawback for these methods is that the approximate Jacobian or its inverse is based on an amalgam of new and old information, so they are unlikely to converge in fewer iterations than Newton’s method. Performance of all these methods will clearly depend on how the Jacobian is changing from iteration to iteration, and how information collected at each function evaluation is used. Memory requirements for Broyden are half that of NKA making it an attractive alternative.

Analytic examinations in [14] show that variants of Anderson Mixing, including NKA, behave like a Krylov method (such as GMRES) when applied to most linear problems. Further, the hypotheses of non-stagnation of GMRES can be removed at the cost of a modest performance penalty.

Our numerical results indicate that Anderson Mixing in the form of NKA found solutions in the PARTISN and Capsaicin codes, for both the reflected and unreflected problem, in the fewest number of function evaluations and the shortest runtimes. These problems contain real material properties and real geometries, and were run at a large computational scale. Our results highlight the strength of this method: regularity, consistency and efficiency. In our results, NKA was shown to bring the norm down to zero smoothly, much as FPI and JFNK do, but with greater efficiency than those methods. Broyden and NKA₋₁, while they at times achieved excellent performance, did not always demonstrate this same smooth convergence behavior and often diverged. Based on these results we feel that NKA may be well-suited to other computational physics problems beyond neutron transport.

6. Acknowledgment

The authors are grateful to Dana Knoll for pointing out the work of Walker and Ni and the work of Fang and Saad and for discussions on nonlinear solvers and their applications.

This work was performed under the auspices of the National Nuclear Security Administration of the US Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396. LA-UR 11-06996

Appendix A. Proof of Theorem 2.3

Proof of Theorem 2.3. From the linearity of the problem $\mathbf{w}_i = \mathbf{A}\mathbf{v}_i$, and from the update rule $\mathbf{x}_{n+1} = \mathbf{x}_n - \mathbf{v}_{n+1}$, we have $\mathbf{x}_n = \mathbf{x}_0 - \sum_{i=1}^n \mathbf{v}_i$, and then

$$\mathbf{r}_n := f(\mathbf{x}_n) = \mathbf{A}\mathbf{x}_n - \mathbf{b} = \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n \mathbf{v}_i,$$

from which we have

$$f(\mathbf{x}_n) - \sum_{i=1}^n z_i^{(n)} \mathbf{w}_i = \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i.$$

We use the above to compute the $n + 1$ residual using the unmodified NKA update

$$\begin{aligned} \mathbf{r}_{n+1} &= \mathbf{A}\mathbf{x}_{n+1} - \mathbf{b} \\ &= \mathbf{A}\mathbf{x}_n - \mathbf{b} - \mathbf{A}\mathbf{v}_{n+1} \\ &= \mathbf{A}\mathbf{x}_n - \mathbf{b} - \mathbf{A} \left[\sum_{i=1}^n \mathbf{v}_i z_i^{(n)} + f(\mathbf{x}_n) - \sum_{i=1}^n \mathbf{w}_i z_i^{(n)} \right] \\ &= \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n \mathbf{v}_i - \mathbf{A} \left[\sum_{i=1}^n \mathbf{v}_i z_i^{(n)} + \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i \right] \\ &= (\mathbf{I} - \mathbf{A}) \left(\mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i \right). \end{aligned}$$

If, for the modified update, $\|\mathbf{w}_n\|_2 \neq 0$ then the n^{th} coefficient is updated as follows

$$z_n^{(n)} \leftarrow z_n^{(n)} \pm \varepsilon \frac{\|f(\mathbf{x}_n) - \sum_{i=1}^n \mathbf{w}_i z_i^{(n)}\|_2}{\|\mathbf{w}_n\|_2} = z_n^{(n)} \pm \varepsilon \frac{\|\mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i\|_2}{\|\mathbf{A} \mathbf{v}_n\|_2}.$$

The n^{th} residual becomes

$$\mathbf{r}_{n+1} = (\mathbf{I} - \mathbf{A}) \left(\mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i \pm \varepsilon \frac{\|\mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i\|_2}{\|\mathbf{A} \mathbf{v}_n\|_2} \mathbf{A} \mathbf{v}_n \right),$$

from which we have

$$\begin{aligned} \|\mathbf{r}_{n+1}\|_2 &\leq \|\mathbf{I} - \mathbf{A}\| \left\| \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i \pm \varepsilon \frac{\|\mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i\|_2}{\|\mathbf{A} \mathbf{v}_n\|_2} \mathbf{A} \mathbf{v}_n \right\| \\ &= (1 + \varepsilon) \|\mathbf{I} - \mathbf{A}\| \left\| \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + z_i^{(n)}) \mathbf{v}_i \right\|_2, \end{aligned}$$

where $\|\cdot\|$ denotes the operator 2-norm. Let $\mathbf{z}^{(n)} = (z_1^{(n)}, \dots, z_n^{(n)})$ denote the solution to the minimization problem

$$\mathbf{z}^{(n)} = \arg \min_{\mathbf{y} \in \mathbb{R}^n} \left\| f(\mathbf{x}_n) - \sum_{i=1}^n \mathbf{w}_i y_i \right\|_2 = \arg \min_{\mathbf{y} \in \mathbb{R}^n} \left\| \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n (1 + y_i) \mathbf{v}_i \right\|_2$$

and let $\mathcal{V}_n = \text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$, then the above gives

$$\|\mathbf{r}_{n+1}\|_2 \leq (1 + \varepsilon) \|\mathbf{I} - \mathbf{A}\| \min_{\mathbf{v} \in \mathcal{V}_n} \|\mathbf{r}_0 - \mathbf{A} \mathbf{v}\|_2. \quad (\text{A.1})$$

If the modification of $z_n^{(n)}$ is not performed, then the factor of $(1 + \varepsilon)$ would simply be one. In either case the above bound on $\|\mathbf{r}_{n+1}\|_2$ given in Eq. (A.1) holds.

We now turn our attention to the question of when the Krylov spaces $(\mathcal{K}_n := \text{span}\{\mathbf{r}_0, \mathbf{A} \mathbf{r}_0, \dots, \mathbf{A}^{n-1} \mathbf{r}_0\})$ are expanding with n . In what follows we assume $\mathbf{r}_0 \neq 0$, i.e. that our initial guess \mathbf{x}_0 is not a solution. Let M be the lowest natural number so that $\mathcal{K}_{M+1} \subseteq \mathcal{K}_M$. Then the vectors $\mathbf{r}_0, \mathbf{A} \mathbf{r}_0, \dots, \mathbf{A}^{M-1} \mathbf{r}_0$ are linearly independent and there is a unique set of coefficients $\alpha_0, \dots, \alpha_{M-1}$ such that

$$\mathbf{A}^M \mathbf{r}_0 = \sum_{i=0}^{M-1} \alpha_i \mathbf{A}^i \mathbf{r}_0.$$

If $\alpha_0 = 0$, then we would have

$$\mathbf{A} \left[\mathbf{A}^{M-1} \mathbf{r}_0 - \sum_{i=0}^{M-2} \alpha_{i+1} \mathbf{A}^i \mathbf{r}_0 \right] = 0,$$

and, because \mathbf{A} is non-singular, this would imply that $\mathcal{K}_M \subseteq \mathcal{K}_{M-1}$ contradicting the definition of M . As such we have

$$\mathbf{r}_0 = \frac{1}{\alpha_0} \left[\mathbf{A}^M \mathbf{r}_0 - \sum_{i=1}^{M-1} \alpha_i \mathbf{A}^i \mathbf{r}_0 \right] = \mathbf{A} \left(\frac{1}{\alpha_0} \left[\mathbf{A}^{M-1} \mathbf{r}_0 - \sum_{i=0}^{M-2} \alpha_{i+1} \mathbf{A}^i \mathbf{r}_0 \right] \right),$$

and hence a solution. This allows us to conclude if we haven't found a solution to the problem within \mathcal{K}_M , then $\mathcal{K}_M \subsetneq \mathcal{K}_{M+1}$.

It now suffices to show that $\mathcal{V}_n = \mathcal{K}_n$. We begin by noting $\mathcal{V}_1 = \text{span}\{\mathbf{v}_1\} = \text{span}\{\mathbf{r}_0\} = \mathcal{K}_1$, and, because \mathbf{A} is of full rank, $\mathbf{w}_1 = \mathbf{A} \mathbf{v}_1 = \mathbf{A} \mathbf{r}_0 \neq 0$. Our inductive hypothesis is that there is some natural number n , e.g. $n = 1$, so that $\mathbf{w}_n \neq 0$ and that for all natural numbers $j \leq n$, $\mathcal{V}_j = \mathcal{K}_j$ and $\mathbf{A} \mathbf{x}_j - \mathbf{b} \neq 0$. That we haven't found a solution in the first n iterations is sufficient to conclude from above arguments that $\mathcal{K}_1 \subsetneq \dots \subsetneq \mathcal{K}_{n+1}$.

The NKA update rule, where the modification to $z_n^{(n)}$ has already been applied (because $\mathbf{w}_n \neq 0$), may be written as

$$\begin{aligned} \mathbf{v}_{n+1} &= \sum_{i=1}^n \mathbf{v}_i z_i^{(n)} + f(\mathbf{x}_n) - \sum_{i=1}^n \mathbf{w}_i z_i^{(n)} \\ &= \sum_{i=1}^n \mathbf{v}_i z_i^{(n)} + \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^n \mathbf{v}_i (1 + z_i^{(n)}) \\ &= \begin{cases} \left[(1 + z_1^{(1)}) \mathbf{r}_0 \right] - \mathbf{A} \mathbf{r}_0 (1 + z_1^{(1)}) & \text{for } n = 1 \quad (\text{recall } \mathbf{v}_1 = \mathbf{r}_0) \\ \left[\sum_{i=1}^n \mathbf{v}_i z_i^{(n)} + \mathbf{r}_0 - \mathbf{A} \sum_{i=1}^{n-1} \mathbf{v}_i (1 + z_i^{(n)}) \right] - \mathbf{A} \mathbf{v}_n (1 + z_n^{(n)}) & \text{for } n \neq 1 \end{cases} \end{aligned}$$

In the case $n = 1$, $\mathbf{v}_2 = (1 + z_1^{(1)})(\mathbf{r}_0 - \mathbf{A} \mathbf{r}_0)$. Our modification ensures that $1 + z_1^{(1)} \neq 0$. If $\mathbf{r}_0 = \mathbf{A} \mathbf{r}_0$, then NKA has found the solution $\mathbf{x}_1 = \mathbf{x}_0 - \mathbf{r}_0$. We have excluded this by assumption and so $\mathcal{V}_2 = \text{span}\{\mathbf{v}_1, \mathbf{v}_2\} = \text{span}\{\mathbf{r}_0, \mathbf{A} \mathbf{r}_0\} = \mathcal{K}_2$ and $\mathbf{w}_2 = \mathbf{A} \mathbf{v}_2 \neq 0$. Our inductive hypothesis holds for $n = 2$.

For the cases $n \neq 1$, we know that $\mathcal{V}_{n-1} = \mathcal{K}_{n-1}$. This implies that any linear combination of $\mathbf{v}_1, \dots, \mathbf{v}_{n-1}$ lies in \mathcal{K}_{n-1} , and so any linear combination of $\mathbf{A} \mathbf{v}_1, \dots, \mathbf{A} \mathbf{v}_{n-1}$ lies within \mathcal{K}_n . We conclude that the quantity in brackets in the last of the preceding lines lies within \mathcal{K}_n . Since \mathbf{x}_n is not the solution, $\text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_{n-1}\} = \mathcal{K}_{n-1} \subsetneq \mathcal{K}_n = \text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$. This is sufficient to conclude that \mathbf{v}_n may be written as the sum $\tilde{\mathbf{v}} + \mathbf{A}^{n-1} \mathbf{r}_0$, where $\tilde{\mathbf{v}} \in \mathcal{V}_{n-1} = \mathcal{K}_{n-1}$. By construction $(1 + z_n^{(n)}) \neq 0$ and so if $\mathbf{v}_{n+1} = 0$, then $\mathcal{K}_{n+1} \subseteq \mathcal{K}_n$ and hence \mathbf{x}_{n+1} will be a solution. Since we assume \mathbf{x}_{n+1} is not the solution, our inductive hypothesis holds for $n + 1$. \square

References

- [1] G. I. Bell, S. Glasstone, Nuclear Reactor Theory, Van Nostrand Reinhold Co., New York, 1970.
- [2] E. E. Lewis, W. F. Miller, Computational Methods of Neutron Transport, American Nuclear Society, Inc., Lagrange Park, IL, 1993.
- [3] A. F. Henry, Nuclear-Reactor Analysis, MIT Press, Cambridge, MA, 1975.
- [4] J. S. Warsa, T. A. Wareing, J. E. Morel, J. M. McGhee, Krylov subspace iterations for k-eigenvalue calculations, Nucl. Sci. Eng. 147 (1) (2004) 26–42.
- [5] D. F. Gill, Y. Y. Azmy, Newton’s method for solving k-eigenvalue problems in neutron diffusion theory, Nucl. Sci. Eng. 167 (2) (2011) 141–153.
- [6] D. F. Gill, Y. Y. Azmy, J. S. Warsa, J. D. Densmore, Newton’s method for the computation of k-eigenvalues in sn transport applications, Nucl. Sci. Eng. 169 (1) (2011) 37–58.
- [7] D. A. Knoll, H. Park, C. Newman, Acceleration of k-eigenvalue/criticality calculations using the Jacobian-free Newton-Krylov method, Nucl. Sci. Eng. 167 (2) (2011) 133–140.
- [8] D. G. Anderson, Iterative procedures for nonlinear integral equations, J. ACM 12 (4) (1965) 547–560.
- [9] N. N. Carlson, K. Miller, Design and application of a gradient-weighted moving finite element code I: In one dimension, SIAM J. Sci. Comput. 19 (3) (1998) 728–765.
- [10] D. A. Knoll, D. E. Keyes, Jacobian-free Newton-Krylov methods: A survey of approaches and applications, J. Comput. Phys. 193 (2) (2004) 357–397.
- [11] C. T. Kelley, Iterative methods for linear and nonlinear equations, Vol. 16 of Frontiers in Applied Mathematics, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995.
- [12] J. Nocedal, S. J. Wright, Numerical Optimization, 2nd Edition, Springer Series in Operations Research and Financial Engineering, Springer, 2006.
- [13] J. E. Dennis, J. J. More, Characterization of superlinear convergence and its applications to quasi-newton methods, Mathematics of Computation 28 (126) (1974) 549–560.
- [14] H. F. Walker, P. Ni, Anderson acceleration for fixed-point iterations, SIAM J. Numer. Anal. 49 (2011) 1715–1735.
- [15] Y. Saad, M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Scientific and Stat. Comp. 7 (3) (1986) 856–869.
- [16] N. N. Carlson, K. Miller, Design and application of a gradient-weighted moving finite element code II: In two dimensions, SIAM J. Sci. Comput. 19 (3) (1998) 766–798.
- [17] P. Ni, Anderson acceleration of fixed-point iteration with applications to electronic structure calculations, Ph.D. thesis, Worcester Polytechnic Institute (2009).
- [18] I. Zavorin, D. P. O’Leary, H. Elman, Complete stagnation of GMRES, Linear Algebra Appl. 367 (2003) 165–183.
- [19] A. Greenbaum, V. Ptak, Z. Strakos, Any nonincreasing convergence curve is possible for GMRES, SIAM J. Matrix Anal. Appl. 17 (3) (1996) 465–469.
- [20] A. Greenbaum, Z. Strakos, Matrices that generate the same Krylov residual spaces, in: Recent Advances in Iterative Methods, Springer, 1994, pp. 95–118.
- [21] H. Fang, Y. Saad, Two classes of multisection methods for nonlinear acceleration, Numer. Linear Algebra Appl. 16 (3) (2009) 197–221.
- [22] R. E. Alcouffe, R. S. Baker, J. A. Dahl, S. A. Turner, R. C. Ward, PARTISN: A time dependent, parallel neutral particle transport code system, Tech. Rep. LA-UR-05-3925, Los Alamos National Laboratory (2005).

- [23] S. J. Plimpton, B. Hendrickson, S. P. Burns, W. McLendon, L. Rauchwerger, Parallel S-n sweeps on unstructured grids: Algorithms for prioritization, grid partitioning, and cycle detection, *Nucl. Sci. Eng.* 150 (3) (2005) 267–283.
- [24] S. D. Pautz, An algorithm for parallel S-n sweeps on unstructured meshes, *Nucl. Sci. Eng.* 140 (2) (2002) 111–136.
- [25] S. Parthasarathy, V. S. A. Kumar, M. V. Marathe, A. Srinivasan, S. Zust, Provable algorithms for parallel generalized sweep scheduling, *J. Parallel Distr. Com.* (2006) 807–821.
- [26] R. S. Baker, K. R. Koch, An S-n algorithm for the massively parallel CM-200 computer, *Nucl. Sci. Eng.* 128 (3) (1998) 312–320.
- [27] J. S. Warsa, A continuous finite element-based, discontinuous finite element method for Sn transport, *Nucl. Sci. Eng.* 160 (2008) 385–400.
- [28] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski, E. T. Phipps, A. G. Salinger, H. K. Thornquist, R. S. Tuminaro, J. M. Willenbring, A. Williams, K. S. Stanley, An overview of the Trilinos project, *ACM T. Math. Software* 31 (3) (2005) 397–423.
- [29] S. C. Eisenstat, H. F. Walker, Choosing the forcing terms in an inexact Newton method, *SIAM J. Sci. Comput.* 17 (1) (1996) 16–32.